AD-A104 417 MASSACHUSETTS INST OF TECH CAMBRIDGE LAB FOR INFORMA--ETC F/G 17/2
MULTIACCESS OF A SLOTTED CHANNEL BY FINITELY MANY USERS,(U)
AUG 81 M G HLUCKYJ, R G GALLAGER N00014-75-C-1183
UNCLASSIFIED LIDS-P-1131 NL

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A104417
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) MULTIACCESS OF A SLOTTED CHANNEL BY FINITELY MANY USERS	5. TYPE OF REPORT & PERIOD COVERED Paper
6. PERFORMING ORG. REPORT NUMBER	LIDS-P-1131
7. AUTHOR(s) M.G. Hluchyj and Robert G. Gallager	8. CONTRACT OR GRANT NUMBER(s) DARPA Contract ONR-N0014-75-C-1183
9. PERFORMING ORGANIZATION NAME AND ADDRESS M.I.T. Laboratory for Information and Decision Systems Cambridge, MA 02139	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Code No. 5T10 ONR Identifying No.049-383
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209	12. REPORT DATE August 1981
	13. NUMBER OF PAGES 7
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Office of Naval Research Information Systems Program Code 437 Arlington, Virginia 22217	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

Accession For
NTIS GRA&I [X]
DTIC TAB [ ]
Unannounced [ ]
Justification
By
Distribution/
Availability Codes
Dist. Special
A

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

SEP 21 1981

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The multiaccess problem as characterized by a finite user population and a time-slotted channel with limited feedback is examined. Results pertaining to the relationship among commonly used performance measures are derived, and insight as to the nature of and difficulty in finding an optimal protocol is given. In addition, three restricted but reasonable classes of protocol are defined and examined.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

410 950

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

MULTIACCESS OF A SLOTTED CHANNEL BY FINITELY MANY USERS*

Michael G. Hluchyj and Robert G. Gallager

Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Abstract

The multiaccess problem as characterized by a finite user population and a time-slotted channel with limited feedback is examined. Results pertaining to the relationship among commonly used performance measures are derived, and insight as to the nature of and difficulty in finding an optimal protocol is given. In addition, three restricted but reasonable classes of protocols are defined and examined.

1. Introduction

A communication problem that has received much attention during the past decade is that of organizing a population of users so that they may efficiently share the resources of a single communication channel. Although various models of the user population and communication channel have been considered, they generally have the following properties. The users are geographically distributed and generate messages (i.e., blocks of digital data to be transported over the channel) in an independent random fashion. The channel is such that only one user at a time can successfully transmit a message, and associated with message transmissions is some form of feedback to the users. This feedback has typically ranged from no feedback (e.g., TDMA) to each individual user determining whether its own message transmissions are successful (e.g., Aloha [1]) to every user determining after some given delay whether there are 0, 1, or $\geq$ 2 messages being transmitted on the channel (e.g., Ethernet [2], Tree [3]).

The problem of organizing or coordinating the transmissions of users for the efficient utilization of the channel is referred to as the multiaccess (or multiple access) problem, and techniques or schemes for coordinating transmissions are called multiaccess protocols. Much of the effort devoted to the multiaccess problem has followed along the lines of protocol development and analysis. The numerous protocols found in the literature reflect not only the fact that there are many different models of the user population and communication channel, but also that the problem of finding the best protocol for any nontrivial model and performance measure is as yet unsolved.

*This research was supported by DARPA Contract ONR-N00014-75-C-1183.

This paper is concerned with the multiaccess problem as characterized by the user-channel model described in Section 2. In Section 3, after specifying several common measures of a protocol's steady-state performance, we derived relationships among the measures and from these we show that the measures are all equivalent. Sections 4, 5, and 6 are concerned with the development of multiaccess protocols. In Section 4 we illustrate that the problem of constructing a protocol that performs optimally is within the *not* well developed Team theory discipline. In Section 5 we describe two restricted but reasonable classes of protocols, the Access Set and Extended Access Set protocols, which allow a classical sequential decision making formulation of the multiaccess problem. In Section 6 we define the class of Window protocols which is a subclass of the Extended Access Set protocols whose state space is finite, and thus amenable to known optimization techniques. The performance of optimal Window protocols are given for the cases of two and three users.

2. The User-Channel Model

We consider a finite population of N users, where the messages generated by each user are incorporated into fixed-length blocks of data called packets. Packet transmissions are synchronized to occur within globally defined time-slots, where the slot size is equal to the time to transmit one packet (a slotted channel). It is assumed that a given slot results in a successful packet transmission if and only if the slot contains exactly one packet. A slot occupied by two or more packets results in a collision where none are successful, requiring each to be retransmitted at a later time. When no packet transmission occurs within a slot, we say the slot *is empty*. As for the channel feedback, immediately following the end of each slot, it is assumed that each user can determine whether the slot contained 0, 1, or $\geq$ 2 packets, corresponding to, respectively, an empty slot, a success, or a collision.

Finally, we assume a homogeneous population of users, where at the beginning of each slot each user independently generates a packet with probability p, but will only accept this packet into its transmission buffer if the buffer is currently empty (the "single buffer" assumption). An unsuccessfully transmitted packet remains in the buffer, and any packets that are generated while the buffer is not empty are assumed lost. Also, a packet entering a buffer at the beginning of a slot may be transmitted in that slot.

This user-channel model is selected for study, not because it necessarily represents any specific practical communication system, but rather because it is indicative of the basic multiaccess problem and thus one in which the fundamental nature of multiaccess may be examined. Other more practical models which incorporate, for example, carrier sense, early detection of collisions, packet reservation, and/or larger transmission buffers, may be viewed as straightforward extensions to this model [4].

3. Steady-State Performance Measures

At an intuitive level, the efficient utilization of the channel basically involves avoiding collisions and avoiding empty slots during busy periods (i.e., when there is at least one packet awaiting transmission). Both of these events correspond to the channel being wasted in the sense that one or more packets are waiting to be "serviced" (i.e., successfully transmitted) and none are actually getting service. The situation where both are eliminated altogether is referred to as perfect scheduling, and represents a desired but unattainable level of performance in a system with both geographically distributed users and randomly generated packets. Indeed, avoiding collisions and avoiding empty slots during busy periods are conflicting goals, and designing an efficient multiaccess protocol essentially involves trading off these two undesirable events.

In this section several common measures of a protocol's steady-state performance are stated and relationships among the measures are derived. From these relationships we show that the measures are all equivalent in the sense that (1) each performance measure may be expressed as a simple function of any one of the others and (2) a protocol selected to be optimal with respect to any one performance measure is optimal with respect to all the others. These results depend only on the user-channel model specified in the previous section (although they are valid independent of the assumed feedback), and on the existence of limits inherent with steady-state statistics.

With the single buffer assumption, a user whose buffer is unable to accept an arriving packet is said to be backlogged and the arriving packet is said to be blocked. Also, each packet in a buffer at the start of a slot is counted as being in the "system" during that slot. With this terminology in mind, consider the following typical steady-state performance measures:

B_u = E[number of backlogged users]

P_b = Pr[an arriving packet is blocked]

B_a = E[number of blocked packet arrivals per slot]

P_s = Pr[successful packet transmission]

N_s = E[number of packets in the system]

D = E[delay of a packet measured in slots from the time the packet enters a buffer until the end of its successful transmission]

Note that under steady-state conditions, P_s is equal to the system throughput (i.e., the fraction of slots containing successful packet transmissions). Through simple probabilistic arguments we have

$$P_b = B_u/N \tag{1}$$

$$B_a = pB_u \tag{2}$$

$$P_s = p(N - B_u) \tag{3}$$

$$N_s = B_u + P_s = pN + (1-p)B_u \tag{4}$$

$$D = N_s/P_s = 1 + [p(N/B_u - 1)]^{-1} \tag{5}$$

where (1) and (2) follow from the independent but homogeneous nature of packet arrivals, (3) follows from the equilibrium condition: E[number of successful packet transmissions per slot] = E[number of unblocked packet arrivals per slot], (4) follows after noting that a user with a buffered packet is only backlogged if it is unable to successfully transmit this packet, and finally (5) follows from an application of Little's result.

Observe from Eqs. (1)-(5) that each performance measure may be written as a function of only p, N, and B_u, and that each such function is monotonic in B_u. Hence it follows that after obtaining any one of the six performance measures, the others are easily determined by evaluating simple algebraic equations. Now, given p and N, it is clear that desirable protocols would minimize B_u, P_b, B_a, N_s, or D, or maximize P_s. Note from Eqs. (1)-(5) that P_s is monotonically decreasing with B_u and that all the other performance measures are monotonically increasing with B_u. Thus a protocol that is optimal with respect to any one of the six performance measures is optimal with respect to the others.

From an analytical point of view these results are significant in that it is often true that a multiaccess protocol is more easily analyzed or optimized with regard to one performance measure than the others. Moreover, we need not be concerned about any trade-off situations where a protocol is optimal with respect to one of the performance measures but not another.

4. Team Protocols

Our emphasis in the remainder of this paper is with the development of multiaccess protocols. We begin by considering the underlying structure of the generic multiaccess protocol for the user-channel model we have specified. At the beginning of each slot, based on its current knowledge of the state of the system, each user with a buffered packet must decide whether to transmit its packet in the slot. A user's knowledge of the "state of the system" may, in general, be based on all the information that is available to it, including the feedback obtained from previous channel outcomes (common information) and the past history of its own packet arrivals and transmission decisions (local information). Moreover, considering the performance measures we have selected, a user's decision to transmit or not is made unselfishly, with the goal being to optimize some global objective function. Such a problem of sequential decision making in an environment of decentralized decision makers with distributed information and a common objective function may be formulated within the framework of Team theory [5].

Although the notion of a dynamic team problem has been around for over 25 years, the class of

problems is of sufficient complexity that little progress has been made toward a general solution technique or even in finding general properties of optimal solutions. Hence its value to the multiaccess problem does not go much beyond a conceptual level.

Without established solution methodologies, one is forced to restrict the scope of feasible solutions to those classes to which known optimization techniques can be applied. In the next two sections of this paper we examine three related subclasses of Team protocols: the Access Set, Extended Access Set, and Window protocols. Each class can be modeled as a Markov decision process [6,7,8], but only with the latter can we generally solve for the optimal protocol.

5. Access Set and Extended Access Set Protocols

An Access Set protocol is of the following structure. At the beginning of each slot, every user follows a common algorithm, based only on common information, that specifies a subset of users which are given permission to access the slot. Each user in this access set with a packet then transmits its packet in the slot. The sequential nature of the process is illustrated in Figure 1 where A(j) is the access set for slot j, T(j) is the subset of users in A(j) which transmit packets in slot j, and C(j) is the common channel observation which for our model corresponds to the ternary channel outcome {0,1,≥2} observed at the end of slot j.

The above structure imposes a form of coordination among the users in which both common and local information are employed in a user's decision to transmit a packet. The channel outcomes are common information and are used in selecting the access set. The local information consists of each user knowing whether it has a packet and thus whether to transmit given that it is in the current access set. The use of the local information is predetermined since by definition a user in the access set is required to transmit if it has a packet. What remains to be specified is the decision algorithm used to determine the access set A(j) at the beginning of each slot j. Since both the algorithm and its inputs are restricted to be common to all users, the problem may be formulated in the context of classical (i.e., nondistributed) sequential decision making [9].

The information available for selecting A(j) are the previous observations C(1),...,C(j-1) and decisions A(1),...,A(j-1) along with the given initial conditions of the system. The decision A(j) may, in general, be a probabilistic function of this past history of the system. However, we require all users to compute the same access set A(j) for each j, and hence any randomization in the decision by the algorithm must have the same outcome at each user. This may be accomplished with the use of identical, precomputed tables of samples from appropriate probability distributions stored at each user; or, for a more practical method, one might consider using a pseudorandom number generator with the same seed at each user. Such a "centralized" structure for randomizing decisions is in reality more general than allowing users to independently randomize their own decisions. To see this, note that one type of centralized structure consists of choosing an independent random decision for each user; in effect each user has knowledge of the other decisions (and thus of the access set) but does not use this knowledge. Hence we may restrict our attention to the class of centrally randomized decisions which includes as a subclass all deterministic decision algorithms. Later when we examine the class of Window protocols, we shall see that there exists an optimal Window protocol whose decision process is deterministic.

The steady-state performance measures examined in Section 3 correspond to the infinite horizon average expected value problem in the sequential decision making nomenclature. Due to their equivalence, any one of the six may be chosen as the reward (cost) function for our problem. One that is easily incorporated into the problem formulation we develop is P_s, the system throughput. Defining the immediate reward

$$r(j) = \begin{cases} 1 \text{ if slot } j \text{ contains a success} \\ 0 \text{ otherwise} \end{cases}$$

we have, assuming the limit and expectation exist,

$$P_s = \lim_{M\to\infty} \frac{1}{M} E\left[\sum_{j=1}^{M} r(j)\right] \qquad (6)$$

where the expectation is conditioned on both the selected decision algorithm and the given initial conditions of the system. Adopting notation from sequential decision making, we shall occasionally refer to the decision algorithm as a policy and the decision A(j) as a control. The problem of interest is that of determining, for any given p and N, a policy which maximizes (6).

To develop a framework for finding an optimal policy, we begin by defining the internal state vector $u(j) = (u_1(j),\ldots,u_N(j))$, where component $u_i(j)$, $i = 1,\ldots,N$, is 1 if user i has a packet at the beginning of slot j and 0 otherwise. For the packet generation process specified in Section 2, it follows that internal state transitions can be modeled by a 2^N-state discrete-time Markov chain where the probabilities governing transition to u(j+1) depend only on u(j) and the control A(j). Note, however, that only the channel outcome C(j) is observed by the decision process, and that this is insufficient to determine the next internal state u(j+1). Nevertheless, since the observation C(j), reward r(j), and transition to u(j+1) depend only on the current internal state u(j) and decision A(j), the optimization problem may be formulated in terms of a partially observable Markov decision process [10]. For such a process it is a standard result that the 2^N-vector $\eta(j)$, where component $\eta_i(j)$ is

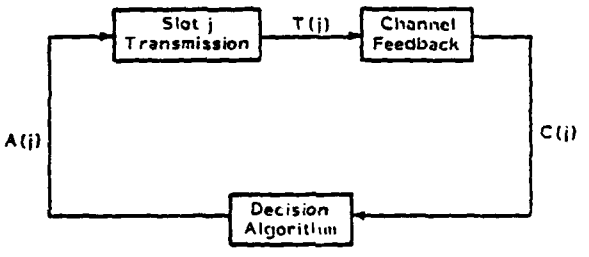

Fig. 1: Access Set protocol structure

the conditional probability of being in internal state i at the beginning of slot j given the previous decisions and observations and the initial conditions of the system, is a sufficient statistic for the complete past history of the process. Moreover, from Bayes' Rule it follows that $\eta(j)$ may be expressed as a function of only $\eta(j-1)$, $A(j-1)$, and $C(j-1)$ and thus computed recursively. Hence, $\eta(j)$ can be viewed as the state of a discrete-time Markov decision process upon which the decision $A(j)$ is based.

The difficulty we now face in determining an optimal policy (i.e., a function mapping $\eta(j)$ into $A(j)$ which maximizes (6)) stems from both the type of performance measure we have selected and the new state space for the process. It is well known that if the state and control spaces are finite, then an optimal policy for an infinite horizon average expected value problem exists and is in the class of stationary deterministic policies*. Moreover, techniques such as Howard's policy iteration algorithm [6] exist for determining such an optimal policy. However, if the state space is allowed to be infinite, then optimal policies may not exist or, when they do exist, they may not be stationary or deterministic [7,8,9]. Now although the internal state $u(j)$ is from a finite state space, the state space corresponding to the new problem is generally infinite. This follows by considering the case where the decision algorithm sets $A(j) = \{1\}$ for all j.

It is of value to note that the class of Access Set protocols may be extended while maintaining the classical sequential decision making formulation of the multiaccess problem. Specifically, one might consider controlling packet transmissions via a time interval mechanism in addition to the access set. That is, a user's packet is transmitted only if the user is in the access set and the packet was generated in some globally defined time interval (or intervals), where both the access set and time interval(s) are computed by each user according to some common algorithm based only on common information. This allows further flexibility in the design of a multiaccess protocol over that of the basic Access Set structure, without precluding a Markovian decision formulation of the problem. Such an extension does, of course, further complicate the already difficult problem of finding an optimal protocol. In the next section, however, we examine a subclass of these Extended Access Set protocols where the Markov decision formulation has a finite state space, and thus one for which an optimal policy can be determined.

Finally, it is worth noting that, aside from variations in the assumed feedback, many of the currently proposed multiaccess protocols may be viewed as being from the general class of Access Set or Extended Access Set protocols. Two simple examples are TDMA and slotted Aloha. With both protocols there is no assumed feedback of common information to the users and so the decision process runs open loop. The decision process is deterministic for TDMA (i.e., access sets contain one member and users are assigned to access sets in a round-robin fashion) and random in the distributed sense for slotted Aloha (i.e., each user independently decides by "flipping a biased coin" whether to belong to the access set). The Urn protocol [11] and especially the Tree protocol [3] are more in line with the type of protocol we have been discussing, since with both, the access set is selected based on the feedback of common information to all users. With the Urn protocol the access set is selected in a centrally randomized fashion. With the Tree protocol the decision process is deterministic, and since, with its frame structure, packets generated during one frame cannot be transmitted until the next, the protocol is a member of the Extended Access Set protocols.

*This is assuming that we have control over the starting state of the system. Without this assumption we would require an additional condition such as that every stationary policy results in a Markov chain containing exactly one irreducible set of states.

6. Window Protocols

The class of protocols discussed in this section use a windowing operation for selecting the access set. Specifically, the N users are ordered (algorithmically speaking) on a circle as illustrated in Figure 2 and the access set is selected by a window that rotates around the circle. That is, at the beginning of each slot, the access set for that slot consists of all users within the window. As for the movement of the window, if a collision occurs, the tail of the window remains fixed and the window size decreases. After an empty slot or a success, the tail of the window advances along the circle to the head of the previous window with the window size possibly changing. Note that the protocol is inherently fair in that for each revolution of the window every user is given the opportunity to successfully transmit one packet. Also, the window approach to selecting the access set simplifies the decision algorithm, since the only decision to be made at the beginning of each slot is the window size. As an indication of it intuitive appeal, this basic windowing concept was independently proposed as an extension to the Tree protocol by Gallager [12] and Urn protocol by Kleinrock and Yemini [11].

The class of Window protocols defined in this section have additional restrictions on how the window size changes and, using a time interval mechanism, one which packets generated by users in the window are allowed to be transmitted. These restrictions actually only occur after a collision, whereupon the operation of the window protocol

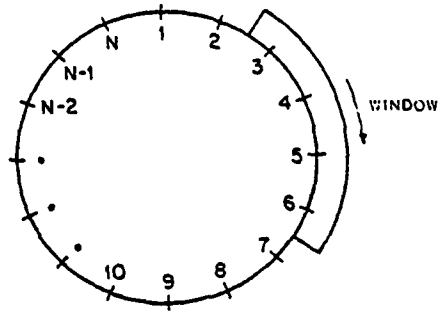

Fig. 2: Access set selection via a windowing operation.

enters a conflict resolution mode. It is instructive to consider first the situation where there are no collisions and then afterward the general case.

Suppose each access set selected by the window results in either a successful transmission or an empty slot. It follows then that each user i will independently have a packet with probability

$$P_i = 1 - (1-p)^{T_i} \qquad (7)$$

where T_i is the positive integer number of slots since user i was last included in the window. If we continually renumber the users so that user 1 is always the first user in the window and user 2 is the next clockwise to 1 and so on, then clearly

$$T_1 \geq T_2 \geq \cdots \geq T_N \qquad (8)$$

so that

$$P_1 \geq P_2 \geq \cdots \geq P_N$$

Hence each user in the window has at least as high a probability of possessing a packet as any user not included in the window. As evidence of the reasonableness of selecting the access set through a windowing operation, it is shown in [4] that the subset of the N users which maximizes the probability of a successful transmission is of the form $\{1,2,\ldots,k\}$ for some $1 \leq k \leq N$ (assuming that each user independently has a packet with probability P_i and $P_1 \geq P_2 \geq \ldots \geq P_N$).

When there is a collision, the protocol enters a conflict resolution mode (steps 2 and 3 in the description below). During this phase a restricted class of users R is specified before the start of each slot. The restriction is that any packet a user generates while in R cannot be considered for transmission until after the user leaves R. This constraint on the protocol is made to maintain a tractable state space, but is also intuitively reasonable since allowing new packets to enter the conflict resolution process can only increase the uncertainty as to which users were originally involved in the collision.

The generic operation of the Window protocol is given in algorithmic form below. For notational convenience we number the users from 0 to N-1 (there is no renumbering in this description as the window changes), and we define the subset of users

$$[i,j] = \begin{cases} i & i = j \\ i,i+1,\ldots,j & i < j \\ i,i+1,\ldots,N-1,0,1,\ldots j & i > j \end{cases}$$

The first line of each of the three steps denotes the control as specified by the window W and restricted class R. Following this is the action taken by the decision algorithm (d.a.) for each of the possible channel outcomes {empty, success, collision}. The process starts at step 1 with no outstanding collisions to resolve, and all additions (+) are computed modulo N.

1) W = [i,j], R = ∅
   if empty or success
     d.a. selects m ∈ {1,2,...,N}
     i ← j+1
     j ← j+m
     go to step 1
   if collision
     d.a. selects k ∈ {i,i+1,...,j-1}
     go to step 2

2) W = [i,k], R = [i,j]
   if empty
     i ← k+1
     d.a. selects k ∈ {i,i+1,...,j-1}
     go to step 2
   if success
     i ← k+1
     d.a. selects k ∈ {i,i+1,...,j}
     go the step 3
   if collision
     j ← k
     d.a. selects k ∈ {i,i+1,...,j-1}
     go to step 2

3) W = [i,k], R = [i,j]
   if empty
     i ← k+1
     d.a. selects k ∈ {i,i+1,...,j}
     go to step 3
   if success
     d.a. selects m ∈ {1,2,...,N}
     i ← k+1
     j ← k+m
     go to step 1
   if collision
     j ← k
     d.a. selects k ∈ {i,i+1,...,j-1}
     go to step 2

Note that at step 2 and step 3 of the description there are, respectively, ≥ 2 and ≥ 1 users in R with packets. At step 1 R is empty.

The positive integer variable T_i, introduced in (7) for the case of no collisions, is a convenient mechanism for tracking the system state upon which the window size decisions are based. It is updated for each user i at the end of every slot following the observation of the channel outcome. The update rules are as follows where τ = 1 + number of slots since R last became nonempty:

(1) i ∉ W, i ∉ R
    $T_i \leftarrow T_i + 1$

(2) i ∈ W, i ∉ R
    if empty or success
      $T_i \leftarrow 1$
    if collision
      no change

(3) i ∉ W, i ∈ R
    if success or collision in step 3 or collision in step 2
      $T_i \leftarrow T_i + \tau$
    otherwise
      no change

(4) i ∈ W, i ∈ R
    if empty or success where i did not transmit
      $T_i \leftarrow \tau$
    if empty or success where i transmitted
      $T_i \leftarrow 1$
    if collision
      no change

Now observe from the protocol description that each user $i \notin R$ independently has a packet with probability P_i given by (7) where T_i is determined from the above update rules. Note, for example, that this is true even though after two successive collisions, those users in the window associated with the first collision but not the second are removed from R. Although this independence does not hold for users that are currently in R, their contribution to the system state is easily characterized by T_i for each $i \in R$, the current value of τ, and whether there are ≥ 2 (when at step 2) or ≥ 1 (when at step 3) users in R with packets. Now suppose that all users have packets, and the window size is set to N and only reduced by 1 after each collision. From this worst case analysis we have $T_i \leq N^2$ for all i. Hence it follows that the state space for the Window protocol is finite, although increasing exponentially with N. Consequently, an optimal policy for the Window protocol exists and is in the class of stationary deterministic policies. Furthermore, it may be shown that this policy exists independent of the system starting state [4].

There are two additional aspects of the Window protocol to be discussed before considering some simple examples. First, note in case (4) of the T_i update rules that to compute the new value for T_i following a success requires the identity of the user that transmitted the packet. This is typically not a problem for a real communication system, but nevertheless represents additional input to the decision process in order for it to keep track of the system state. Second, also from case (4) note that to maintain the ordering of the T_i's as specified in (8) (assuming the renumbering of users), and thus in a sense the fairness of the protocol, requires that users at times be reordered on the circle. Both of these complications result from the packet transmission restrictions that stem from R.

Two User Case

As indicated, the system state under the Window protocol may be characterized by the vector $(T_1,\ldots,T_N)$ when at step 1 of the protocol description and by $(T_1,\ldots,T_N)$, R, and τ when at steps 2 and 3. This leads to a cumbersome notation for the state space which is unnecessary for the simple case of two users. Here τ does not enter the picture, and when the system is at step 2 each user is known to have a packet (which we denote by setting $T_1 = T_2 = \infty$) and similarly when at step 3 one user $i \in \{1,2\}$ is known to have a packet (denoted by $T_i = \infty$). To further simplify the state space, we dynamically renumber the users so that user 1 always corresponds to the first user in W. Through an exhaustive search of all possible window size decisions and channel outcomes, we find that the system will be in one of the following four states after at most two slot-times:

$$S_1 = (1,1) \qquad S_2 = (\infty,\infty)$$
$$S_3 = (\infty,1) \qquad S_4 = (2,1)$$

A stationary deterministic policy P assigns to each state $i \in \{1,2,3,4\}$ a window size $w_i \in \{1,2\}$. It is easily verified from the Window protocol description that $w_2 = w_3 = 1$; thus leaving 4 feasible policies to consider. The transition probabilities and expected immediate rewards for each combination of state and window size decision are polynomials in p that are easily determined. Either through an exhaustive search or an application of Howard's policy iteration algorithm [6], an optimal policy $P^* = [w_1^*, w_2^*, w_3^*, w_4^*]$ for the case of two users is found to be

$$P^* = \begin{cases} [2,1,1,2] & \text{for } 0 \leq p \leq s \\ [1,1,1,1] \text{ or } [2,1,1,1] & \text{for } s < p \leq 1 \end{cases} \quad (9)$$

where $s \approx 0.3473$ is the solution to $0 = 1 - 3s + s^3$ for $s \in [0,1]$. The performance of this optimal policy as characterized by the system throughput P_s is given by

$$P_s = \begin{cases} p(2 - p^2 + p^3)/(1 + p^2 + p^3) & \text{for } 0 \leq p \leq s \\ 1 - (1-p)^2 & \text{for } s < p \leq 1 \end{cases}$$

From Section 3 we obtain

$$D = 1 + N/P_s - 1/p$$

and thus the optimal performance in terms of the average delay D is given by

$$D = \begin{cases} 1 + p(3 + p)/(2 - p^2 + p^3) & \text{for } 0 \leq p \leq s \\ 1 + 1/(2 - p) & \text{for } s < p \leq 1 \end{cases}$$

These results are plotted in Figure 3, along with the performance of perfect scheduling.

Note from (9) that for $p \leq 0.3473$ a window size of 2 is used except following a collision, whereupon the window size is reduced to 1 for the next two slots, allowing each user to transmit alone. When p exceeds 0.3473 the control switches to a constant window size of 1; this, of course, is just TDMA.

Three User Case

For N = 3, the state space consists of 23 states and an optimal policy may be found that switches, as p varies from 0 to 1, among six different policies (the sixth corresponds to TDMA). The delay performance of these six policies are plotted in Figure 4, where the switching points are indicated by vertical lines. Although a detailed analysis of the three user case may be found in [4], we summarize a few of the more important results here. First, if there results a collision after the window size is set to 2, the protocol operates as in the case of N=2. If there results a collision after the window size is set to 3, the window is reduced to size 1, allowing the first user to access the channel, and then, if the first user sends a packet, the window is increased to size 2 allowing the remaining two users to access the channel. Finally, for each of the step 1 states, the window size switches from 3 to 2 to 1 as p increases from 0 to 1. This type of behavior is expected in a system that must trade off the undesirable effects of both collisions and empty slots during busy periods.

7. Conclusions

We have shown that the multiaccess problem, even for a relatively simple user-channel model, is complex; having its underlying theoretical

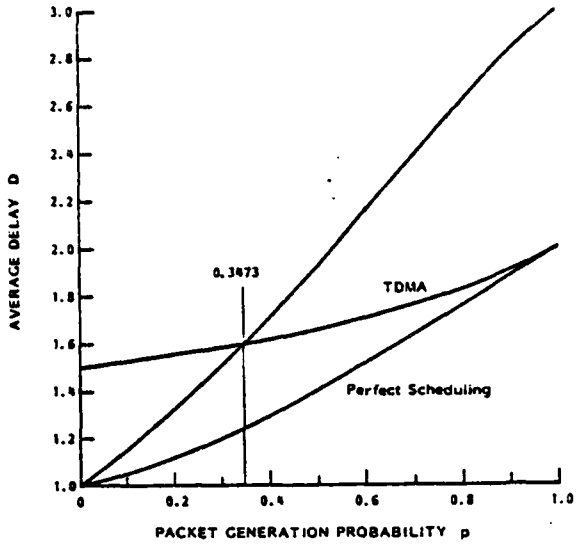

Fig. 3: Optimal perf. of Window protcol for N=2

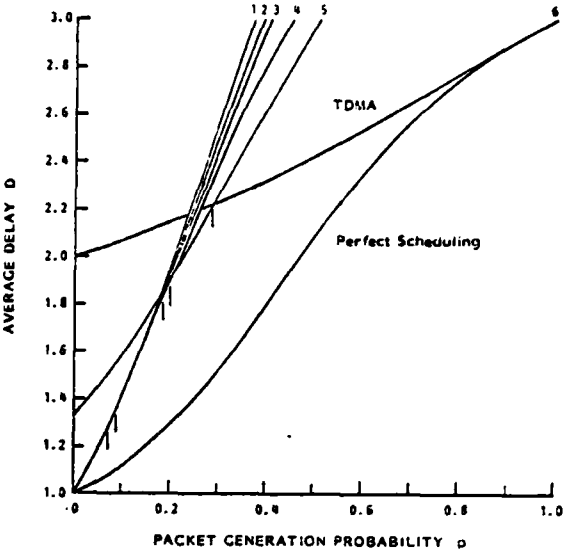

Fig. 4: Optimal perf. of Window protocol for N=3

structure in an area that is not well understood. Although by restricting the protocol structure we were able to formulate the problem of finding an optimal protocol in a more classical context, it nevertheless remained a difficult problem. This was well illustrated by the infinite state space associated with the Access Set and Extended Access Set protocols, and the exponential growth in the state space with population size in the case of the Window protocols.

An alternative to the approach of searching for an optimal protocol is to determine theoretical bounds on protocol performance and then search for a heuristic protocol whose performance is in some sense close to the derived bounds. The results on steady-state performance measures in Section 3 could be useful here, both in the determination of bounds and the analysis of heuristic protocols.

References

[1] N. Abramson, "The ALOHA System - Another Alternative for Computer Communications," AFIPS Conf. Proc., vol. 37, pp. 281-285, 1970.

[2] R. Metcalfe, D. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks," Comm. ACM, vol. 19, pp.395-404, July 1976.

[3] J. Capetanakis, "Generalized TDMA: The Multi-Accessing Tree Protocol," IEEE Trans. Comm., vol. COM-27, pp. 1476-1484, October 1979.

[4] M. Hluchyj, "Multiple Access Communication: The Finite User Population Problem," Ph.D. Thesis, Dept. of Elec. Eng. and Comp. Sci., M.I.T., Cambridge, MA, Oct. 1981.

[5] J. Marschak, R. Radner, Economic Theory of Teams, Yale Univ. Press, New Haven, CT, 1972.

[6] R. Howard, Dynamic Programming and Markov Processes, MIT Press, Cambridge, MA, 1960.

[7] C. Derman, Finite State Markovian Decision Decision Processes, Acad. Press, N.Y., N.Y., 1970.

[8] S. Ross, Applied Probability Models with Optimization Applications, Holden-Day, San Francisco, CA, 1970.

[9] D. Bertsekas, Dynamic Programming and Stochastic Control, Academic Press, New York, NY, 1976.

[10] R. Smallwood, E. Sondik., "The Optimal Control of Partially Observable Markov Processes over a Finite Horizon," Oper. Res., vol. 21, pp. 1071-1088, 1973.

[11] L. Kleinrock, Y. Yemini, "An Optimal Adaptive Scheme for Multiple Access Broadcast Communication," ICC Conf. Proc., pp. 7.2.1-7.2.5, June 1978.

[12] R. Gallager, "Conflict Resolution in Random Access Broadcast Networks," Proc. of AFOSR Workshop on Comm. Theory and Appl., pp. 74-76, Sept. 1978.